(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号

特開平5－257811

(43)公開日　平成５年(1993)10月８日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　12/10		Ｈ　7232－5Ｂ		
9/46	３４０	Ｆ　8120－5Ｂ		

審査請求　未請求　請求項の数８（全 24 頁）

(21)出願番号　特願平5－3937

(22)出願日　平成５年(1993)１月13日

(31)優先権主張番号　特願平4－26040

(32)優先日　平４(1992)１月16日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地

(72)発明者　斎藤　光男
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　浅野　滋博
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　申　承昊
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(74)代理人　弁理士　則近　憲佑

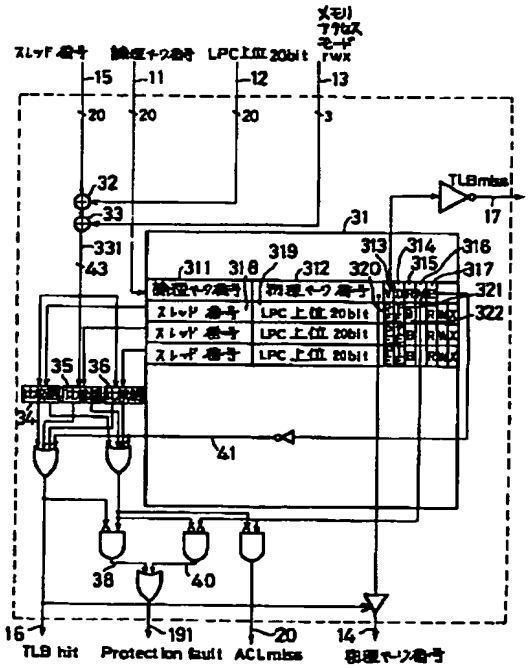
最終頁に続く

(54)【発明の名称】　メモリ管理装置

(57)【要約】

【目的】　複数の領域に分割され、その中を複数のスレッドが実行するアドレス空間の管理を行ない、メモリアクセスに対してアクセス権の制御を適切に行なう。

【構成】　論理アドレスから物理アドレスへの変換を行なう機能を有するもので、ＴＬＢ（ページテーブルのキャッシュ）３１中に、論理アドレスに対するアクセスを許すスレッドの番号と領域の番号と許すアクセスの方法を組にしたものであるアクセスコントロールリストを複数格納し、これらのアクセスコントロールリストのスレッド番号と領域番号の組のうち、現在実行中のスレッドの番号１５とそのスレッドが現在実行しているプログラムの存在している領域の番号１２を組み合わせたものに一致するものを比較器３４～３６で検出し、ここで一致するものが検出されるとＴＬＢヒット１６として出力するようにしている。



1

【特許請求の範囲】

【請求項１】　複数のスレッドを起動して仮想空間に割り付けられたプログラムを並行して実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうメモリ管理装置であって、

現在プログラムを実行しているスレッドの番号を記憶する手段と、

前記論理アドレスをどのスレッドがアクセス可能かを示す情報を記憶する手段と、

前記スレッド番号を持つスレッドが前記論理アドレスへアクセスしようとする場合に前記アクセス可能かを示す情報を用いて前記アクセスをしようとしているスレッドにアクセスの権限があるか否かを検証する検証手段とを具備し、

この検証手段によりアクセスの正当性が保証された場合に前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とするメモリ管理装置。

【請求項２】　検証手段において、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号の一致を調べる際に、スレッド番号の全部又は一部をマスクする手段を具備することを特徴とする請求項１記載のメモリ管理装置。

【請求項３】　検証手段において、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号について所定の論理演算を行い、論理演算結果が真であった場合にスレッド番号が一致したと見做す手段を具備することを特徴とする請求項１記載のメモリ管理装置。

【請求項４】　複数の領域に分割された仮想空間に割り付けられたプログラムを実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうメモリ管理装置であって、

前記論理アドレスをアクセスするプログラムの置かれている領域がどれであるかを検知する手段と、

前記論理アドレスが前記仮想空間中のどの領域に置かれているプログラムからアクセス可能かを表す情報を記憶する手段と、

プログラム実行中に前記論理アドレスへのアクセス命令が出現した場合に前記アクセス可能かを表す情報を用いて前記アクセス命令が置かれている領域からのアクセスが正当か否かを検証する検証手段とを具備し、

この検証手段によりアクセスの正当性が保証された場合に、前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とするメモリ管理装置。

【請求項５】　検証手段において、どの領域からのアクセスが可能かを示す情報のなかの領域番号と、アクセスしようとしているプログラムの置かれている領域の領域番号の一致を調べる際に、領域番号の全部又は一部をマスクする手段を具備することを特徴とする請求項４記載

2

のメモリ管理装置。

【請求項６】　検証手段において、どの領域からのアクセスが可能かを示す情報のなかの領域番号と、アクセスしようとしているプログラムの置かれている領域の領域番号について所定の論理演算を行い、論理演算結果が真であった場合に領域番号が一致したと見做す手段を具備することを特徴とする請求項４記載のメモリ管理装置。

【請求項７】　仮想空間に割り付けられたプログラムを実行する計算機に設けられ、アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段を有するメモリ管理装置であって、

前記アドレス変換テーブル中のアドレス対の各々に対して複数個のアクセス保護情報を記憶する手段と、

前記複数個のアクセス保護情報を参照し、実行中のプログラムの仮想空間内の位置からのあるいはプログラムを実行中の実行主体による前記指示されたアドレスへのアクセスが許可されるか否かを検証する手段と、

前記検証結果を出力する手段とを具備したことを特徴とするメモリ管理装置。

【請求項８】　仮想空間に割り付けられたプログラムを実行する計算機に設けられ、

アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段と、

前記アドレス変換テーブルの一部を置くキャッシュメモリを有するメモリ管理装置であって、

前記アドレス変換テーブル中に設けられ、アドレス対の各々に対して複数で可変個のアクセス保護情報を記憶する第１の記憶手段と、

前記キャッシュメモリ中に設けられ、アドレス対に各々に対して複数で固定個のアクセス保護情報を記憶する第２の記憶手段と、

前記第２の記憶手段中の複数のアクセス保護情報を検証する検証手段と、

前記検証手段で検証できなかった場合に、前記キャッシュメモリ中の第２の記憶手段の内容のみを、前記アドレス変換テーブル中の第１の記憶手段の内容の一部と入れ換える手段を具備したことを特徴とするメモリ管理装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、メモリアクセスを制御するメモリ管理装置に関するものである。

【０００２】

【従来の技術】近年、ネットワーク技術の発達や並列計算機技術の発展により、例えばサーバ・クライアント型プログラミングのように、複数のプログラムがデータを

共有し、協調しあって処理を進める方式が広がって来た。

【０００３】この方式では、それぞれのプログラムは、仮想記憶の技術を用いて別々のアドレス空間におかれているため、データを共有するためにはオペレーティングシステム（ＯＳ）を介してデータのやりとりをしなければならず、ＯＳのオーバヘッドにより処理速度が遅くなってしまうという欠点があった。

【０００４】そこで、一つのアドレス空間内に、実行主体であるスレッドを複数走らせ、スレッド間のデータの共有を、ＯＳのオーバヘッドなしでおこなうという方式もおこなわれている。

【０００５】この方式は、もともとスレッド間の保護を考えていないものであるため、あるスレッドが使っているデータを他のスレッドから保護しようとすると、オーバヘッドの大きい方法を採らなければならない。

【０００６】すなわち、アドレス空間ごとの保護機能を利用してスレッド毎に別々のアドレス空間を割り当て、それぞれのアドレス空間に、そのスレッドからアクセスできる部分だけを置くことになる。これはかなりオーバヘッドが大きく実用的ではない。以上が第１の問題点、つまり１つのプログラム中でスレッドごとに許されるアクセスの種類を変えることができないという問題点である。

【０００７】さらにスレッド毎にアドレス空間を作るためには、それぞれの仮想空間ごとにページテーブルを用意しなければならず、同じプログラムは別々のアドレス空間内でも同じ位置に置かなければならないが、こうすると複数のスレッドで共通にアクセスできる領域についても、アクセスが許されるスレッド全てが、同じ論理－物理アドレスの変換情報を重複してページテーブル中に持たなければならないため、ページテーブルとして使うメモリが無駄である。

【０００８】もちろん、複数のスレッドで共通にアクセスできる領域の、許されるアクセス方法が同じであれば、それらのスレッドで、一つのページテーブルの部分を共有することは出来、ページテーブルとして使うメモリの無駄をなくすことが出来るが、この場合でもＴＬＢ（Translation Look-aside Buffer）のようなページテーブルのキャッシュを使っている場合には、無駄が発生し、アドレス変換の平均速度を低下させる。

【０００９】その理由を説明すると、まず、ページテーブルのキャッシュの１エントリは、ページテーブルの情報とスレッド番号を組にして持っているため、スレッドが違えば、ページテーブルの情報が全く同じでも、異なるキャッシュエントリを使うことになる。

【００１０】したがって、同じ論理－物理アドレス対をなすが、スレッド番号の異なるものがキャッシュの複数のエントリを占有してしまうことになり、ページテーブルのキャッシュ内に格納できる物理アドレスの種類が少

なくなってしまい、キャッシュの大きさが小さくなったのと同等に、アドレス変換の速度の低下が起こる。

【００１１】さらに、複数のページテーブルに置かれた同じページに対する情報を管理しなければならないため、ページングの処理が複雑になるという問題点がある。例えば、複数のスレッドが同じ物理ページをそれぞれの仮想アドレス空間にマッピングしている時に、そのページをページアウトしようとすると、その物理ページをマッピングしている全ての仮想空間を探して、そのページに対応するページテーブルのエントリを無効化する処理が必要になる。

【００１２】また、ページインする時も、ページインしようとしている物理ページを共有する全てのアドレス空間の、そのページに対応するページテーブルのエントリを直さなければならない。これらの処理時間が平均アドレス変換時間に加算される。以上が第２の問題点、つまり、ページテーブルや、ページテーブルのキャッシュのメモリが無駄になり、かつ、管理が複雑になり非効率であるという問題点である。

【００１３】さらに、別の方法として一つのアドレス空間を複数の領域に分けて、スレッドがどの領域のプログラムを実行しているかによって、そのスレッドの持つ権限を変える、ということが、アプリケーションによっては有効である。例えば、領域Ａのデータを変えられるのは、別のある領域Ｂにあるプログラムを実行しているスレッドに限る（領域Ｃにあるプログラムを実行しているスレッドは領域Ａのデータを変えられない）ようにすれば、領域Ａのデータを、一定の条件を保ちながら変更することが可能となる。領域Ａにデータベースのデータ、領域Ｂにデータベースのプログラムでそのデータへのアクセスルーチンを入れておけば、データベース処理に関して、データベースのデータを保護しつつ、且つ、ＯＳを介することなく、そのデータをアクセスできるため、高速なアクセスが可能となる。

【００１４】しかし、従来のメモリ管理装置（例えばインテル社の８０４８６プロセッサ）では、一つのスレッドが実行するプログラムの範囲をセグメント方式等を利用することによって限定し、かつ、コールゲートを利用して異なるアクセスレベルに移ることによってアクセス権限を変えることは出来たが、アクセスレベルは、順序関係を持ち、レベルが上がった場合にはそのレベル以下でアクセス可能な領域は全てアクセスできるというものであり、かつハードウェアによってレベルの数を限定されてしまうため、アプリケーションプログラムとＯＳとカーネルといったような順序関係をもったアクセス権の制御しか利用できないなど柔軟性に欠けるという欠点があった。

【００１５】以上が第三の問題点、つまり一つのアドレス空間中に複数の領域に分けてプログラムやデータ等を配置した場合、領域毎に許されるアクセスの種類を柔軟

に変えることができないという問題点である。

【００１６】

【発明が解決しようとする課題】このように、従来のメモリ管理装置で、一つのアドレス空間内を実行する複数の実行主体（スレッド）の間で保護を行おうとすると、同じ物理ページをマッピングしているアドレス空間が多数あるため、ページングを行なう場合には、全てのアドレス空間を探して、そのページがマッピングされているものを見つけ、それらに対してページングの処理を行なわなければならない等、オーバヘッドが大きいという第１の問題点があった。

【００１７】さらに、同じ物理アドレスに対してページテーブル中の多くのエントリを占有するために、他の物理アドレスに対するアドレス変換を阻害することになり、また、ページテーブルのエントリが複数のプログラムから共有されているか否かに関わらず無効化されてしまい、エントリを満たすための不必要な処理が行なわれる欠点があった。また、複数のプログラムが持つアクセスの権利の正しさを同時に検査しなければならない場合に、係る検査にかかる処理が複雑になってしまうという欠点があった。つまり、ページテーブルやページテーブルキャッシュに同じ情報が複数置かれてしまい、メモリが無駄になり、キャッシュの効果が薄れ、非効率であるという第２の問題点があった。

【００１８】また、一つのアドレス空間を複雑の領域に分けて、スレッドがどの領域のプログラムを実行しているかによって、そのスレッドの権限を変える、ということは、従来のメモリ管理装置では、柔軟に行なえないという第３の問題点があった。

【００１９】本発明は、上記事情に鑑みなされたもので、複数のプログラムで共有されるアドレス空間の管理を行ない、メモリアクセスに対してアクセス権限の制御を適切に行なうことが出来、しかもアドレス変換時間を延長することもないメモリ管理装置を提供することを目的とする。

【００２０】

【課題を解決するための手段】上記第一の問題点を解決する本発明に係るメモリ管理装置は、複数のスレッド（実行主体）を起動して仮想空間に割り付けられたプログラムを並行して実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうものであって、現在プログラムを実行しているスレッドの番号を記憶する手段と、前記論理アドレスをどのスレッドがアクセス可能かを示す情報を記憶する手段と、前記スレッド番号を持つスレッドが前記論理アドレスへアクセスしようとする場合に前記アクセス可能かを示す情報を用いて前記アクセスをしようとしているスレッドにアクセスの権限があるか否かを検証する検証手段とを具備し、この検証手段によりアクセスの正当性が保証された場合に前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とする。

【００２１】上記第二の問題点を解決する本発明に係るメモリ管理装置は、仮想空間に割り付けられたプログラムを実行する計算機に設けられ、アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段を有するものであって、前記アドレス変換テーブル中のアドレス対の各々に対して複数個のアクセス保護情報を記憶する手段と、前記複数個のアクセス保護情報を参照し、実行中のプログラムの仮想空間内の位置からのあるいはプログラムを実行中のスレッド（実行主体）による前記指示されたアドレスへのアクセスが許可されるか否かを検証する手段と、前記検証結果を出力する手段とを具備したことを特徴とする。

【００２２】上記第三の問題点を解決する本発明に係るメモリ管理装置は、複数の領域に分割された仮想空間に割り付けられたプログラムを実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうものであって、前記論理アドレスをアクセスするプログラムの置かれている領域がどれであるかを検知する手段と、前記論理アドレスが前記仮想空間中のどの領域に置かれているプログラムからアクセス可能かを表す情報を記憶する手段と、プログラム実行中に前記論理アドレスへのアクセス命令が出現した場合に前記アクセス可能かを表す情報を用いて前記アクセス命令が置かれている領域からのアクセスが正当か否かを検証する検証手段とを具備し、この検証手段によりアクセスの正当性が保証された場合に、前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とする。

【００２３】

【作用】第１の発明によれば、メモリ管理装置中にアドレス空間中の各領域（位置）に対してスレッドごとのアクセス可否を示す情報があり、更に現在実行中のスレッドを識別できるので、現在実行中のスレッドがアクセス先の領域（位置）にアクセスできる権限があるかの検証を行うことで、同じアドレス空間中の異なるスレッドごとのアクセス制御が可能となる。

【００２４】ここで、スレッドとは実行主体であり、ＣＰＵの割り合て単位である。つまりＯＳはスレッド番号ごとにＣＰＵのレジスタ等のメモリ以外に現在の実行を制御する情報を管理する。ＯＳ中にはこれらの情報を退避するメモリ領域があり、スレッドを切り替える時はこの領域の内容とＣＰＵのレジスタ等とを入れかえることで実行の切替えを実現する。コンテクストとも呼ばれる。

【００２５】第２の発明によれば、メモリ管理装置中のページテーブル（論理アドレスと物理アドレスの対応表）の各エントリに対応して複数のアクセス保護情報を保持することができるので、従来の方式のように単一の

アクセス情報しか保持できないもので構成するのと比較してページテーブル及びそのキャッシュのメモリ使用量が大巾に減り、効率が向上する。

【００２６】第３の発明によれば、メモリ管理装置中にアドレス空間の各領域（位置）に対してどの領域からアクセスできるかを示す情報があり、更に現在実行中の領域を識別できるので、現在実行中のプログラムがアクセス先の領域（位置）にアクセスできる権限があるかの検証が行え、同じアドレス空間中の異なる領域ごとのアクセス制御が可能となる。

【００２７】

【実施例】

［実施例１］以下、本発明の一実施例を図面に従い説明する。

【００２８】図１は、同実施例のメモリ管理装置の全体構成を示すものである。この場合、メモリ管理装置１は、アドレス変換装置２、ＴＬＢ(Translation Look-aside Buffer）チェック装置３、スレッド番号記憶部４、ＯＲゲート５からなるProtection fault信号生成回路から構成されている。

【００２９】そして、メモリ管理装置１は、図示しないＣＰＵから論理ページ番号、すなわち論理アドレスの上位２０ビット１１、現在実行中の命令が存在する論理アドレスつまり、一命令前のプログラムカウンタＰＣ（以下ＬＰＣと略す）の上位２０ビット１２、ＣＰＵのメモリアクセスのモードの３ビット１３を取り込み、物理ページ番号、すなわち物理アドレスの上位２０ビット１４を出力するようにしている。この場合、現在実行しているスレッド番号をスレッド番号記憶部４に保存するとともに、論理アドレスの上位２０ビット１１、ＬＰＣの上位２０ビット１２、ＣＰＵのメモリアクセスモードの３ビット１３およびスレッド番号記憶部４に記憶されている現在実行しているスレッド番号が、アドレス変換装置２とＴＬＢチェック装置３に同時に与えられ、２つのモジュールが同じアドレス変換を並行して行なうようになる。

【００３０】ここで、アドレス変換装置２は、ページテーブル（主記憶上に存在するアドレス変換テーブル）を使って論理ページ番号を物理ページ番号に変換する。一方、ＴＬＢチェック装置３は、ＴＬＢ内にキャッシュされている論理アドレスと物理アドレスの対応の中に目的とする論理アドレスが存在するかを調べ、見つかった時にはそれに対応する物理ページ番号を出力するとともに、アドレス変換装置２に対してＴＬＢヒットを１６を送る。

【００３１】そして、アドレス変換装置２がＴＬＢヒット１６を受けとった時は、論理ページ番号を物理ページ番号に変換するアドレス変換操作を中止する。また、ＴＬＢチェック装置３が、目的とする論理ページ番号をＴＬＢ内で見つけられなかった時は、ＴＬＢmissの信号線１７をアサートし、アドレス変換装置２によって変換が行なわれるのを待ち、最終段である第３段目のアドレス変換テーブルのエントリが見つかったところで、その内容をＴＬＢに保存し、その中から物理ページ番号を出力する。アドレス変換装置２において、目的の論理アドレスに対応するページテーブルエントリが存在しないことが判明した場合は、ページフォルト１８を発生してＣＰＵに割り込む。また、両モジュールのいずれかにおいて、アドレス変換の最中に、そのアドレスに対するアクセスが許されないということが判明した時には、プロテクションフォルト１９を発生しＣＰＵに割り込む。

【００３２】図２は、ＴＬＢチェック装置３の概略構成を示すものである。この場合、ＴＬＢ３１のエントリの構成要素は、論理ページ番号３１１、物理ページ番号３１２、ページの属性を示す５個のフラグ３１３、３１４、３１５、３１６、３１７と、３組のスレッド番号３１８、ＬＰＣ上位２０ビット３１９、これらスレッドに対して許されるアクセスの種類であるＲＷＸ（Ｒは読み込み権，Ｗは書き込み権，Ｘは実行権を示す）のパーミッションの組からなっている。以下の説明では、スレッド番号３１８、ＬＰＣ上位２０ビット３１９、ＲＷＸのパーミッションの組を、アクセス・コントロール・リスト、略してＡＣＬと呼ぶことにする。このＡＬＣは、どのプログラムがどの論理ページ上のコードを実行している時に、これからアクセスしようとする論理ページに対してどのようなアクセスが許されているかということを示すものである。

【００３３】この場合、ＴＬＢエントリにはさらにフラグ領域があり、フラグ（Ｖ）３１３はＴＬＢのエントリが有効であるか否かを示すバリッドフラグ、フラグ（Ｄ）３１４はページが変更されたかを示すダーティフラグ、フラグ（Ｒ）３１５はページがアクセスされたかを示すリファレンスドフラグ、フラグ（Ｍ）３１６はページに対するＡＣＬがＴＬＢ内に存在する３組以外に存在するか否かを示すモアフラグ、フラグ（Ｅ）３１７はページのＡＣＬのチェックを行なうかどうかを示すイネーブルフラグＥからなっている。また、ＡＣＬは、スレッド番号３１８、ＬＰＣ上位２０ビット３１９に対して３個のフラグ３２０、３２１、３２２を持っている。この場合、フラグ（ＰＥ）３２０はスレッド番号の比較を行なうか否かを示し、フラグ（ＴＥ）３２１はＬＰＣ上位２０ビットによる比較を行なうか否かを示し、フラグ（Ｂ）３２２はスレッド番号の比較を行なう際にスレッド番号の一致のみを調べるか、あるいはスレッド番号の大小関係を調べるかを示している。

【００３４】しかして、外部からの論理ページ番号の２０ビット１１によりＴＬＢ３１のエントリを特定すると、スレッド番号の２０ビット１５、ＬＰＣの上位２０ビット１２、ＣＰＵのアクセスモードの３ビット１３が連結器３２、３３で連結され、４３ビットの出力３３１

が得られ、これとＴＬＢ３１内にキャッシュされているＡＣＬとの比較が３個の比較器３４、３５、３６を使って行われるようになる。ここで、ＬＰＣの上位２０ビット１２は、後述する仮想空間の領域の領域番号の一例である。

【００３５】図３は、比較器３４、３５、３６の概略構成を示している。この場合、比較器３４、３５、３６は、まったく同じものなので、ここでは比較器３４についてのみ述べると、加算器３４１、３４２、アンドゲート３４３～３４６、オアゲート３４７、３４８、ノアゲート３４９、３５０から構成されている。

【００３６】そして、このような比較器３４は、ＡＣＬのフラグ（ＰＥ）３２０を０にすることで、スレッド番号によるアクセス条件のチェックを省くことができ、全てのスレッドに共通のアクセス許可を与えることができるようにしている。また、フラグ（ＴＥ）３２１を０にした場合は、ＬＰＣの値によるアクセス条件の判定を行なわず、ＡＣＬのエントリに書かれているスレッド番号をもつスレッドは、ＬＰＣの値に関わらず、そのページへのアクセス許可を与えることができるようにしている。また、フラグ（Ｂ）３２２を１にした場合は、現在実行されているスレッド番号がＡＣＬのプログラム番号よりも大きいか等しい時に、スレッド番号によるアクセス条件を満たしているものとし、フラグ（Ｂ）３２２が０の時は、二つのスレッド番号が等しい時に限り、スレッド番号によるアクセス条件を満たすようにしている。また、フラグ（Ｂ）３２２を１にすることにより、スレッド番号をリング保護におけるレベルとして使い、スレッド番号が大きいスレッドに対してより強い権利を与えるようにしている。

【００３７】そして、比較器３２は、２本の出力信号線３５１、３５２を有している。ここで、出力信号線３５２の出力は、現在実行されているスレッドのスレッド番号とＡＣＬ内に書かれているスレッド番号がマッチし、且つ一つ前のプログラムカウンタＰＣ（ＬＰＣ）の上位２０ビットがＡＣＬ内のプログラムカウンタＬＰＣの上位２０ビットの値にマッチした場合にアサートされる。すなわち、ＣＰＵによるメモリアクセスの条件がこのＡＣＬの条件と一致していることを示すようにしている。また、出力信号線３５１の出力は、出力信号線３５２の出力がアサートされ、且つＣＰＵによるメモリアクセスが、ＡＣＬによって許可されている種類のものであった時にアサートされる。つまり、出力信号線３５１の出力がアサートされれば、メモリアクセスは許可されることになる。

【００３８】そして、図２に戻って、各比較器３４、３５、３６の出力信号線３５１の出力の論理和をとった結果が１であれば、ＴＬＢエントリ内の３つのＡＣＬのうちの一つによってアクセスが許されたことになるので、物理ページ番号１４をイネーブルし、ＴＬＢヒット信号線１６がアサートされる。また、各比較器３４、３５、３６の出力信号線３５２の出力の論理和をとった結果が１であり、出力信号線３５１の出力の論理和をとった結果が０の場合は、ＴＬＢエントリ内の３つのＡＣＬのうち、現在のメモリアクセスの条件に一致するものが存在するが、ＣＰＵによるメモリアクセスのモードがこのＡＣＬによって許されていない種類のものであったことを示している。この場合には、信号線３８がアサートされ、プロテクションフォルト１９１がアサートされる。

【００３９】また、比較器３４、３５、３６の出力信号線３５２の出力の論理和をとった結果が０であり、ＴＬＢエントリのモアフラグ３１６が０の場合は、現在のメモリアクセス条件に一致するＡＣＬが存在しないことを示している。この場合は信号線４０がアサートされ、プロテクションフォルト１９１がアサートされる。

【００４０】ＴＬＢエントリのイネーブルフラグ３１７の値が０のときは、信号線４１がアサートされＴＬＢヒット信号線１６がアサートされる。よってＡＣＬのチェックに関係なく物理ページ番号１４を出力する。また、ＴＬＢがヒットしなかった場合には、前述のようにＴＬＢmiss信号線１７がアサートされる。同時に動作しているアドレス変換装置２によって物理ページ番号が判明する可能性が残されているので、その結果を待つと同時にＡＣＬがチェックされる。

【００４１】さらにＴＬＢにはヒットしたがＡＣＬのエントリにはヒットせず、かつモアビット（Ｍ）３１６が１の場合はＡＣＬmiss信号線２０がアサートされる。同時に動作しているアドレス変換装置２によってＡＣＬがチェックされる。次に、図４はアドレス変換装置２の概略構成を示している。

【００４２】この場合、第１段目のアドレス変換装置２１、第２段目のアドレス変換装置２２、第３段目のアドレス変換装置２３により構成されている。そして、このようなアドレス変換装置２は、スレッド番号の２０ビット１５、ＬＰＣの上位２０ビット１２、ＣＰＵのアクセスモードの３ビット１３と論理ページ番号の２０ビット１１を取り込み、論理ページ番号の２０ビット１１に対応する物理ページのページテーブルエントリ２４を出力するようにしている。また、ＴＬＢヒット１６、プロテクションフォルト１９１を受け取った場合はアドレス変換を中止するようにもしている。さらに、アドレス変換の途中で、目的とする論理ページに対応する物理ページが存在するが、アクセスのモードが許されない種類のものであることが判明した場合にはプロテクションフォルト１９２を出力し、アドレス変換の途中で目的とする論理ページに対応する物理ページが存在しないことが判明した場合にはページフォルト１８を出力するようにしている。第１段目のアドレス変換装置２１では、ルートポインタ２５と論理ページ番号の上位７ビット１１１を使って第１段ページテーブルのエントリを決定する。

【００４３】第２段アドレス変換装置２２では、第１段アドレス変換装置２１によって判明した第２段ページテーブルの先頭アドレス２６と論理ページ番号の上位８ビット目から上位１３ビット目までの７ビットによって示されるページテーブル内のエントリ番号１１２を使って第２段ページテーブルのエントリを決定する。

【００４４】第３段アドレス変換装置２３では、第２段アドレス変換装置２２によって判明した第３段ページテーブルの先頭アドレス２７と論理ページ番号の下位６ビットで示されるページテーブル内のエントリ番号１１３を使って第３段ページテーブルのエントリを決定する。図５は、第１段目アドレス変換装置２１および第２段目のアドレス変換装置２２の概略構成を示している。

【００４５】この場合、アドレス変換装置２１、２２は、スレッド番号の２０ビット１５、ＬＰＣの上位２０ビット１２、ＣＰＵのアクセスモードの３ビット１３を連結させた４３ビットの信号線２１１、ＴＬＢヒット１６、論理ページ番号の連続する７ビットの部分１１１（１１２）、ルートポインタ２５（先頭アドレス２６）を取り込み、このうちの論理ページ番号の連続する７ビットの部分１１１（１１２）、ルートポインタ２５（先頭アドレス２６）をアドレス合成器２１２で合成することにより、この論理ページ番号に対するページテーブルエントリ２１３のアドレスを得るようにしている。

【００４６】ページテーブルエントリ２１３は、次レベルページテーブルへのポインタ２１３１、３種類のフラグ２１３２～２１３４、３組のＡＣＬ、そのページテーブルエントリに対するＡＣＬが３組を越える場合に残りのＡＣＬの置かれている場所を指すポインタ２１３５によって構成されている。

【００４７】ここで、３種類のフラグ２１３２～２１３４のうちバリッドフラグ（Ｖ）２１３２は、ページテーブルエントリが有効であるかを示している。モアフラグ（Ｍ）２１３３は、そのページに対応するＡＣＬがページテーブルエントリ内に存在する３組以外に存在するか否かを示している。イネーブルフラグ（Ｅ）２１３４は、ＡＣＬのチェックを行なうか否かを示している。

【００４８】ページテーブルエントリの位置が決まると、バリッドフラグ２１３２の内容を調べ、この値が０であれば、ページフォルト１８を出力し、アドレス変換中止装置２２７を起動し、アドレス変換を中止する。

【００４９】また、４３ビットの信号線２１１の入力と３個のＡＣＬの内容２１４、２１５、２１６を３つの比較器２１７、２１８、２１９を使って同時に比較し、この比較の結果によりアクセスが許可されることが判明した場合は、信号２２０がアサートされ、次レベルページテーブルへのポインタ２１３１の値がイネーブルされる。また、アクセスが不許可であることが判明した場合は、信号２２１がアサートされプロテクションフォルト１９２が出力される。

【００５０】一方、現在のアクセス条件に一致するＡＣＬが３つのものの中から見つからなかった場合は、モアフラグ２１３３を調べる。そして、モアフラグ２１３３が０であった場合には、ＡＣＬが他にないことを示しているので、アクセスの許可がなされないことになり、信号２２３がアサートされ、プロテクションフォルト１９２を出力するようになる。また、モアフラグ２１３３が１であった場合は他にもＡＣＬが存在していることになるので、信号２２４がアサートされ、ＡＣＬ切替装置２２５を使って、次ＡＣＬへのポインタ２１３５によって指示されている次ページテーブルエントリ２２６のＡＣＬの組に切替えて再び比較を行なうようになる。

【００５１】この場合、ポインタ２１３５によって指示されているＡＣＬの組の中に現在のアクセス条件に一致するものがさらになかった場合には、モアビット２２６２が１であるかを調べ、１であれば、再びＡＣＬ切替装置２２５によって次ＡＣＬへのポインタ２２６１によって指される次ページテーブルエントリのＡＣＬの組に切替えて再比較するようになる。なお、ＴＬＢヒット１６を受け取った場合はアドレス変換中止装置２２７により変換処理を中止するようになる。図６は、第３段目のアドレス変換装置２３の概略構成を示している。

【００５２】この場合、アドレス変換装置２３は、スレッド番号の２０ビット１５、ＬＰＣの上位２０ビット１２、ＣＰＵのアクセスモードの３ビット１３を連結させた４３ビットの信号線２３１、ＴＬＢヒット１６、論理ページ番号の下位６ビットの部分１１３、先頭アドレス２７を取り込み、このうちの論理ページ番号の下位６ビットの部分１１３と先頭アドレス２７とをアドレス合成器２３２で合成することにより、この論理ページ番号に対応するページテーブルエントリ２３３のアドレスを得るようにしている。

【００５３】ページテーブルエントリ２３３は、物理ページ番号２３３１、５種類のフラグ２３３２～２３３６、３組のＡＣＬ、そのページテーブルエントリに対するＡＣＬが３組を越える場合に残りのＡＣＬの置かれている場所を指すポインタ２３３７によって構成されている。

【００５４】ここで、３種類のフラグ２１３２～２１３４のうちバリッドフラグ（Ｖ）２３３２はページテーブルエントリが有効であるかを示している。ダーティフラグ（Ｄ）２３３３はページが変更されたか否かを示している。リファレンスドフラグ（Ｒ）２３３４はページに対してアクセスがなされたか否かを示している。モアフラグ（Ｍ）２３３５はページに対応するＡＣＬがページテーブルエントリ内に存在する３組以外に存在するか否かを示している。そして、イネーブルフラグ（Ｅ）２３３６はＡＣＬのチェックを行なうか否かを示している。

【００５５】ページテーブルエントリの位置が決まると、バリッドフラグ２３３２の内容を調べ、この値が０

であれば、ページフォルト１８を出力し、アドレス変換中止装置２４７を起動しアドレス変換を中止する。

【００５６】また、４３ビットの信号線２３１の入力と３個のＡＣＬの内容２３４、２３５、２３６を３つの比較器２３７、２３８、２３９を使って同時に比較し、この比較の結果によりアクセスが許可されることが判明した場合は、信号２４０がアサートされ、物理ページ番号２３３１とフラグ２３３２～２３３６の組と３つのＡＣＬの内容２３４～２３６を連結した１９２ビットの出力２３３８がイネーブルされる。また、アクセスが不許可であることが判明した場合は、信号２４１がアサートされ、プロテクションフォルト１９２が出力される。

【００５７】一方、現在のアクセス条件に一致するＡＣＬが３つのものの中から見つからなかった場合は、モアフラグ２３３５を調べる。モアフラグ２３３５が０であった場合には、ＡＣＬが他にはないことを示しているので、アクセスの許可がなされないことになり、信号２４２がアサートされ、プロテクションフォルト１９２を出力するようになる。また、モアフラグ２３３５が１であった場合は他にもＡＣＬが存在していることになるので、信号２４３がアサートされ、ＡＣＬ切替装置２４５を使って、次ＡＣＬへのポインタ２３３７によって指示されている次ページテーブルエントリ２４６のＡＣＬの組に切替えて、再び比較を行なう。

【００５８】この場合、ポインタ２１３５によって指示されているＡＣＬの組の中に現在のアクセス条件に一致するものがさらになかった場合には、モアビット２２６２が１であるかを調べ、１であれば、再びＡＣＬ切替装置２４５によって次ＡＣＬへのポインタ２４６１によって指されるＡＣＬの組に切替えて再び比較を行なうようになる。

【００５９】図７は、ＡＣＬの組を一回切替えた後の比較によってアクセスが許可されることが判明した場合の状態を示すもので、図６と同一部分には同符号を付して示している。この場合、ページテーブルエントリ２３３の物理ページ番号２３３１とフラグ２３３２～２３３６の組とページテーブルエントリ２４６の３つのＡＣＬの内容２４７～２４９を連結した１９２ビットの出力２４６２がイネーブルされるようになる。

【００６０】本実施例では、メモリ管理装置中のページテーブルのエントリ中に記載するＡＣＬはＬＰＣの上位２０ビット固定であるが、この場合、アクセス元の領域の大きさが固定される。領域の大きさを可変または複数の領域をまとめて１つのＡＣＬで表現できれば柔軟性が増す。その場合には図１２のように１つのＡＣＬ中にさらにマスク領域３２２（最大でＬＰＣ上位ビット数と同じビット数）を設け、このマスクに従って比較を行うようにすればよい。ＡＣＬ中のスレッド番号についても同様なマスク領域をさらに付加すれば複数のスレッドをグループでまとめたアクセス制御が可能となる。

【００６１】また、図３においてアクセス権の検証を行う比較器の構成でスレッド番号及びＬＰＣの比較を行うのに、減算器３４１、３４２を用いているが、ここを論理演算器に置きかえてもよい。この場合、一例として減算器３４２のかわりにＡＮＤ演算器とする。そしてアドレス空間の上位をＯＳなどのシステムに、下位をアプリケーションのプログラムに配置する構成とする。ＬＰＣ上位２０ビット３１９に１６進数で８０００００が入っているとＬＰＣ上位２０ビット１２の値が１６進数で８０００００以上の場合、ＡＮＤ演算器の出力は１になる。つまり、システムプログラムからのアクセスは許可され、アプリケーションのプログラムからのアクセスは拒否される。このように種々の論理演算器及びその組合せに置きかえることが可能になる。

【００６２】また、本実施例では、論理アドレスから物理アドレスを求めるのに３段のアドレス変換装置を利用し、かつ、各段においてＡＣＬのチェックを行っているが、最終段のみＡＣＬチェックを行い第１段、第２段においてはＡＣＬのチェックを行わない構成も取り得る。

［実施例２］

【００６３】以上第１の実施例では、従来の第１、第２、第３の問題点を解決する手段をまとめて示したが、ここで、上記実施例のうち、従来の第１の問題点を解決する手段を取り出し、第２の実施例として説明する。まずアクセス権限の判定に関して説明する。

【００６４】図８において、論理アドレス空間８０の中には、複数のスレッド８１、８２が存在している。ページテーブル８５は図２の一部を取り出したものである。スレッド番号１を持つスレッド８１が、あるアドレスのメモリをアクセスする命令８３を実行しようとする。ここで命令８３は、１６進数表記で１２３４番地のデータをロードしてくることを意味する。まず、ページテーブル８５の中から、アクセスしようとするアドレスが存在するページに対応するエントリが選ばれる。次に、スレッド番号検出部８６によって、現在実行中のスレッド番号が検出され、このスレッド番号に一致するスレッド番号を持った保護情報のエントリが選択される。

【００６５】このエントリの中に書いてある、そのスレッドに許されるアクセスの種類は、アクセスの種類検出部８４に入力される。８４は、実行されつつあるアクセスの種類が、スレッドに許されるアクセスの種類に含まれるか否かを検出し、その結果を物理アドレス／プロテクション・フォルト信号出力部８７に送る。８７は、アクセスが許される場合は、アクセスされようとしている論理ページの番号に対応する物理ページの番号と、論理ページ内のオフセットを合わせて物理アドレスとして出力する。アクセスが許されない場合は、プロテクションフォルト信号を出力する。

【００６６】図８のページテーブル８５の内容によると、論理ページ番号が１であるページは、スレッド１か

らは読み書きが出来るが実行は出来ず、スレッド２からは読み書きが出来ないが実行は出来る、というようになっている。このように、一つのアドレス空間内を走る複数のスレッドについて、別々のアクセス権限を与えることが出来る。

【００６７】さらに詳しく説明すると、スレッドがある論理ページをアクセスしようとすると、ページテーブルの中の、前記論理ページに対応するエントリが選択され、まず、前記論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かを、フラグによってページフォルト検出部８８が検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【００６８】それ以外の場合は、そのエントリの中に複数存在する、アクセスを許すスレッドの番号のフィールドのうちで、現在実行中のスレッド番号と一致するものが存在するかどうかを、比較器を使って検出する。比較器を複数用意して並列に比較すると、比較のための時間が一回比較する時間で済むことになる。

【００６９】一致するフィールドが存在しない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致するフィールドがあった場合は、スレッドが行なおうとしているアクセスの種類が、そのスレッドに許されているアクセスの種類に含まれているかどうかを、比較器を用いて検出する。この比較は、スレッド番号の一致を検出することと同時に行なうことで比較時間を短縮することが出来る。含まれていることが検出されなければプロテクションフォルトとして、割り込み信号を出力する。含まれていることが検出されたときは、アクセスされつつある論理ページに対応する物理ページ番号を出力する。

【００７０】ＯＳがスケジューリングによって、一つのアドレス空間の中で実行するスレッドを切り替える場合は、メモリ管理装置内のスレッド番号記憶部に保存されている、実行中のスレッドの番号を変更する。これにより、これ以降のアドレス変換時は、変更後のスレッド番号と、その論理ページをアクセス出来るスレッド番号の一致を調べることになる。　一つのアドレス空間の中を実行する複数のスレッドに関するページテーブルは、一つにまとまっているため、ＯＳがページングを行なう時は、前記ページテーブルの該当するエントリの、論理－物理アドレスの対応の部分の、物理アドレスの部分を書き換え、そのエントリに対応する物理ページが存在するかどうかを示すフラグを更新するだけでよい。

［実施例３］さらに、第１の実施例のうち、従来の第３の問題点を解決する手段を取り出し、第３の実施例として説明する。まず、アクセス権限の判定に関して説明する。

【００７１】図９において、論理アドレス空間９０は、複数の領域９１、９２、９３に分割されている。ページテーブル９６は図２の一部を取り出したものである。領域９１の中のプログラムを実行中に、あるアドレスのメモリをアクセスする命令９４を実行しようとすると、まず、ページテーブル９６の中から、アクセスしようとするアドレスの存在するページに対応するエントリが選ばれる。次に、領域番号検出部９７によって、現在実行されつつあるプログラムの存在する領域の領域番号を検出し、この領域番号に一致する領域番号を持った保護情報のエントリが選択される。

【００７２】このエントリの中に書いてある、その領域内のプログラムに対して許されるアクセスの種類は、アクセスの種類検出部９５に入力される。９５は、実行されつつあるアクセスの種類が、その領域内のプログラムに対して許されるアクセスの種類に含まれるか否かを検出し、その結果を物理アドレス／プロテクション・フォルト信号出力部９８に送る。９８は、アクセスが許される場合は、アクセスされようとしている論理ページの番号に対応する物理ページの番号と、論理ページ内のオフセットを合わせて物理アドレスとして出力する。アクセスが許されない場合は、プロテクションフォルト信号を出力する。

【００７３】図９のページテーブル９６の内容によると、論理ページ番号が１０であるページは、領域１からは読み書きは出来ないが実行は出来、領域２からは読み書きは出来るが実行は出来ない、となっている。このように、一つの論理アドレス空間内の複数のプログラムに、別々のアクセス権限を与えることが出来る。

【００７４】なお、論理アドレス空間を分割する領域の大きさは、１バイト単位、ページ単位、１メガバイト単位など、どのような単位でも構わない。また、ページの大きさも、好きなように決めても構わない。

【００７５】さらに詳しく説明すると、スレッドが実行しているプログラムが置かれている領域によって、アクセス権限を変える場合は、実行中の命令の存在するアドレスつまり１つ前のプログラムカウンタの上位何ビットかを、現在実行されているプログラムの領域の番号として使用する。

【００７６】まず、アクセスされようとしてる論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かを、フラグによってページフォルト検出部９９で検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【００７７】次に、アクセスを行なうプログラムが置かれている領域の領域番号と、ページテーブル中の論理－物理アドレスの各対応毎に記憶されている、アクセスを許す領域の番号が一致するかどうかを、比較器を使って検出する。一致する領域番号がない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致する番号があった場合は、これから実行されようとしてい

るアクセスの種類が、その領域から許されるアクセスの種類に含まれているかどうかを比較器を用いて検出する。含まれていることが検出されなければプロテクションフォルトとして割り込み信号を出力する。含まれていることが検出された時は、アクセスされつつある論理ページに対応する物理ページの番号を出力する。

［実施例４］さらに第１の実施例のうち、従来の第３の問題点を解決する部分の変形例を第４の実施例として説明する。まず、アクセス権限の判定に関して説明する。

【００７８】図１０において、論理アドレス空間１００は、複数の領域１０１、１０２、１０３に分割されている。領域１０１の中のプログラムを実行中に、あるアドレスのメモリをアクセスする命令１０４を実行しようとすると、領域番号検出部１０８によって現在実行されつつあるプログラムの存在する領域の領域番号を検出する。次に、その領域番号を利用してＡＣＬテーブル１０７の対応する領域番号のＡＣＬエントリを選択する。次にアクセスしようとするアドレスの存在するページ番号を持った保護情報のエントリが選択される。

【００７９】このエントリの中に書いてあるアクセス先ページに対して許されるアクセスの種類はアクセスの種類検出部１０５に入力される。１０５は実行されつつあるアクセスの種類が、そのアクセス先のページに対して許されるアクセスの種類に含まれるか否かを検出してもし許されない場合はプロテクションフォルト信号を出力する。

【００８０】図１０のＡＣＬテーブル１０７の内容によると、領域番号が１である領域からは論理ページ番号１０へは読み書きはできないが実行のみ出来、論理ページ番号１１へは読み書きができるが、実行は出来ない、となっている。このように、一つの論理アドレス空間内の複数のプログラムに、別々のアクセス権限を与えることが出来る。

【００８１】なお、論理アドレス空間を分割する領域の大きさは、１バイト単位、ページ単位、１メガバイト単位など、どのような単位でも構わない。また、ページの大きさも、好きなように決めても構わない。

【００８２】更に詳しく説明すると、スレッドが実行しているプログラムが置かれている領域によって、アクセス権限を変える場合は、実行中の命令の存在するアドレス、つまり１つ前のプログラムカウンタの上位何ビットかを、現在実行されているプログラムの領域の番号として使用する。

【００８３】まず、アクセスされようとしてる論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かをページテーブル１０６のフラグによってページフォルト検出部１１０が検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【００８４】次に、アクセスを行なう先の論理ページ番号と、ＡＣＬテーブル１０７中の、アクセスを行なうプログラムが置かれている領域の領域番号毎に記憶されている、アクセスを許す論理ページの番号が一致するかどうかを、比較器を使って検出する。一致する論理ページ番号がない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致する番号があった場合は、これから実行されようとしているアクセスの種類が、そのページに対して許されるアクセスの種類に含まれているかどうかを比較器を用いて検出する。含まれていることが検出されなければプロテクションフォルトとして割り込み信号を出力する。プロテクションフォルト信号が出力されないことで含まれていることが検出された時は、アクセスされつつある論理ページに対応する物理ページの番号を物理アドレス出力部１０９から出力する。

【００８５】また、図１０におけるＡＣＬテーブル１０７中のＡＣＬエントリはアクセス先の論理ページ番号とそのページに対するアクセスの種類の対で記載されているが、アクセス先の領域番号とその領域に対するアクセスの種類の対で記載してもよい。その場合、図１１のようになりＡＣＬテーブル１０７が１１２のように変更され、新たに設ける領域番号検出部１１１によって、アクセス先のアドレスに相当する領域番号を検出して、ＡＣＬテーブル１１２の対応するエントリを選択する。あとの処理は図１０の場合と同様である。

【００８６】なお、第３の実施例では領域２中の論理アドレスに対して領域１中のプログラムからアクセスできるかという検証方法を、本実施例では領域１中のプログラムを実行中に領域２中の論理アドレスに対してアクセスできるかという検証方法を採っているが、ページテーブルをハッシュを用いて構成することにより両者を融合した検証方法も採り得る。また、本実施例でページテーブル１０６とＡＣＬテーブル１０７を別々に設けているのは、メモリ容量の節約のためである。

【００８７】このように上記第１～第４の実施例によれば、論理アドレス空間を共有する複数のスレッドが、複数の領域におかれているプログラムを実行することによって行なうメモリアクセスに対して、ＴＬＢやページテーブル中の一つのエントリにＡＣＬを複数個格納して対応させるようにしたので、同じ物理アドレスに対して多くのエントリを占有することがなく、しかも複数のエントリが論理アドレスと物理アドレスの対を重複して持つことがなくなる。従って、ページテーブル占有するメモリの量が減少する。また、ＴＬＢ（ページテーブルのキャッシュ）中に格納されている論理アドレスの範囲が広くなり、ＴＬＢのヒット率を向上することができる。また、ＴＬＢやページテーブル中の複数のＡＣＬのそれぞれと、現在実行中のスレッドのスレッド番号と実行されているプログラムの置かれている領域番号の組の比較を同時に行なうことができる。このため、アドレス変換の時間は、従来のメモリ管理装置と同等にできる。

【００８８】また、上記実施例によれば、ＴＬＢを検索中に発生したミスを、アドレスのミスと、ＡＣＬのミスに分けることが可能となり、また、ＴＬＢのアドレスを格納する部分と、ＡＣＬを格納する部分を、独立にページテーブル中の情報と入れ換えることができる。これにより、ＴＬＢの一つのエントリ内に、そのエントリに対応する論理ページに対するＡＣＬをすべて格納できない場合でも、当該エントリを無効化せず、ＡＣＬの部分のみを入れ換えることができる。従って、ＴＬＢミスの処理にかかる時間を短縮することができる。これは特に、論理アドレスを共有するスレッドあるいはプログラムの置かれている領域の数が、一つのエントリ中に格納できる数より多い場合に有効である。

【００８９】なお、上記実施例では、ページテーブルが保持する複数個のＡＣＬについて、スレッド番号と領域番号の組と、その組に許可するアクセスの種類が一つづつ組み合わさせている場合を示したが、複数のスレッド番号と領域番号の組に対して、共通に許可するアクセスの種類を一つだけ記憶するようにしても良いし、複数のスレッド番号に対して領域番号と許可するアクセスの種類を各一つ共通に記憶しても、複数の領域番号に対してスレッド番号と許可するアクセスの種類を各一つ共通に記憶しても良い。スレッドあるいは領域毎に個別に許されるアクセス方法を記憶させればきめ細かいアクセス権制御ができ、一方、アクセスを許すスレッドあるいは領域の集合に共有させて許されるアクセス方法を一つ記憶させればアクセス権制御のきめ細かさは多少落ちるが、ページテーブルとして使用されるメモリの量が節約できる。

【００９０】また、上記実施例では保護の最小単位を物理ページ単位としたが、メモリ管理装置が管理できる大きさであればどのような大きさが単位であっても、またセグメントのように可変量であっても良い。

【００９１】その他、ＴＬＢが保持するページテーブルの一つのエントリに複数個のＡＣＬを保持するという第２の発明は、どのように情報が格納されている場合にも適用可能であることはもとより、ＡＣＬの内容として、上記のスレッド番号と、領域番号であるＬＰＣ上位２０ビットの組以外でも適用可能であり、また、各信号発生回路および制御回路の構成、各種制御信号の与え方なども種々変更して実施することができる。また実施例４のようにページテーブルとＡＣＬテーブルを別々に分けることにより、全体のメモリ量を減らすこともできる。

【００９２】また、上記実施例によれば、アクセスの権利の正しさを検証するため、スレッドを識別する番号だけでなく、スレッド毎のメモリアクセスの仕方を示すアクセス情報のエントリを設けることが可能となり、メモリアクセスに対してきめ細かいアクセス権制御を行なうことが可能になる。

【００９３】また、上記実施例によれば、メモリアクセス毎にスレッド番号あるいは領域番号の全部又は一部をマスクしてグループ化し一致の検出を行なうことが可能となる。これはメモリアクセスに対してグループ化したアクセス権制御を行なうようにできる。そのため、効率的なアクセス権制御を行うことができる。

【００９４】また、上記実施例によれば、スレッド番号あるいは領域番号の一致を検出するだけでなく、大小関係が成立する等、所定の論理演算結果が真となる場合を一致が得られたとすることができ、メモリアクセスに対してプログラム間の又はスレッド間の相対的なアクセス権制御を行なうようにできる。

【００９５】

【発明の効果】本発明は、以上に説明したように構成されているので、以下に記載するような効果を奏する。

【００９６】まず、ページテーブル内の論理－物理アドレスの対応に対して、どのスレッドがその論理アドレスに対してアクセス可能かという情報を記憶する手段を設けることにより、一つのアドレス空間内を実行する複数のスレッド同士の保護を、ページテーブルのために使用するメモリの量を無駄にすることなく、行なうことが出来る。

【００９７】更に、１つの論理－物理アドレス対に対して複数個のアクセス保護情報を記憶すれば、一つの論理－物理アドレスの対応を記憶するために使われるメモリの量が少なくなるため、ＴＬＢに置くことができる論理－物理アドレスの対応の数が増え、キャッシュのヒット率が上がるため、平均のアドレス変換時間が短縮される。

【００９８】そして、一つのアドレス空間を複数の領域に分割し、ページテーブル内の一つの論理－物理アドレスの対応に対して、どの領域に存在するプログラムを実行した時に、その論理アドレスに対してアクセス可能かという情報を１個以上記憶する手段を設けることにより、一つのアドレス空間内に存在する複数のプログラムに対して別々のアクセス権限を与えることが出来る。これにより、あるデータをアクセスするためには、必ず、ある特定のアクセスプログラムを介してしか行われないようにすることができる。

【００９９】また、一つのアドレス空間を複数の領域に分割し、かつ複数のスレッドを実行させる場合に、ページテーブル内の論理－物理アドレスの対応に対して、どのスレッドがどの領域に置かれているプログラムを実行している時に、その論理ページに対してアクセス可能かという情報を記憶する手段を設けることにより、上で述べた効果が同時に達成される。

【０１００】上記のアクセス可能か否かを示す複数の情報を、並行して検証すれば、一つの論理－物理アドレスの対応に対して複数の保護情報があっても、アクセス権のチェックは保護情報が１個しかない場合と同じ時間で行なうことができる。

21

【０１０１】尚、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号の一致を調べてアクセスの正当性を検証する際に、スレッド番号の一部をマスクする機能を具備すれば、スレッド番号のグルーピングが可能となるため、複数のスレッドに対して同じ権限を与えたい場合にページテーブルに置かなければならない情報の量とチェック時間を節約することが出来る。

【０１０２】同様に、検証の際に、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号についての論理演算の結果に基づきスレッド番号が一致したと見做す機能を具備することによって、スレッド番号のグルーピングが可能となるため、複数のスレッドに対して同じ権限を与えたい場合にページテーブルに置かなければならない情報の量とチェック時間を節約することが出来る。このようなグルーピングは領域番号についても同様に実現できる。

【０１０３】さらに、ＴＬＢ（ページテーブルのキャッシュ）を検索中に発生したミスを、アドレスのミスと、ＡＣＬ（保護情報）のミスに分けることが可能となり、また、ＴＬＢのアドレスを格納する部分と、ＡＣＬを格納する部分を、独立にページテーブル中の情報と入れ換えることができる。これにより、ＴＬＢの一つの論理ページに対応するエントリ内に、その論理ページに対するＡＣＬをすべて格納できない場合でも、当該エントリを無効化せず、ＡＣＬの部分のみを入れ換えることができる。従って、ＴＬＢミスの処理にかかる時間及び保護情報ミスの処理にかかる時間を短縮することができる。

【０１０４】このように本発明によれば、一回のアドレス変換に要する時間は従来と同等で、ページテーブルのヒット率が向上し、ミス処理にかかる時間を短縮できることから、全体としてアドレス変換時間を短縮できることになり、また、複数のスレッドが複数の領域に分割されたアドレス空間を実行する時の保護を、スレッド番号と領域番号によってＯＳの介在なしで柔軟に管理することが出来るようになる。

【図面の簡単な説明】

【図１】　本発明のメモリ管理装置の一実施例を示す概略構成図。

【図２】　図１に示す実施例に用いられるＴＬＢチェック装置の概略構成を示す図。

【図３】　図２に示すＴＬＢチェック装置に用いられる比較器の概略構成を示す図。

【図４】　図１に示す実施例に用いられるアドレス変換装置の概略構成を示す図。

22

【図５】　図４に示すアドレス変換装置に用いられる第１段および第２段目のアドレス変換装置の概略構成を示す図。

【図６】　図４に示すアドレス変換装置に用いられる第３段目のアドレス変換装置の概略構成を示す図。

【図７】　図６に示すアドレス変換装置の動作を説明するための図。

【図８】　本発明のメモリ管理装置の第２の実施例を示す概略構成図。

【図９】　本発明のメモリ管理装置の第３の実施例を示す概略構成図。

【図１０】　本発明のメモリ管理装置の第４の実施例を示す概略構成図。

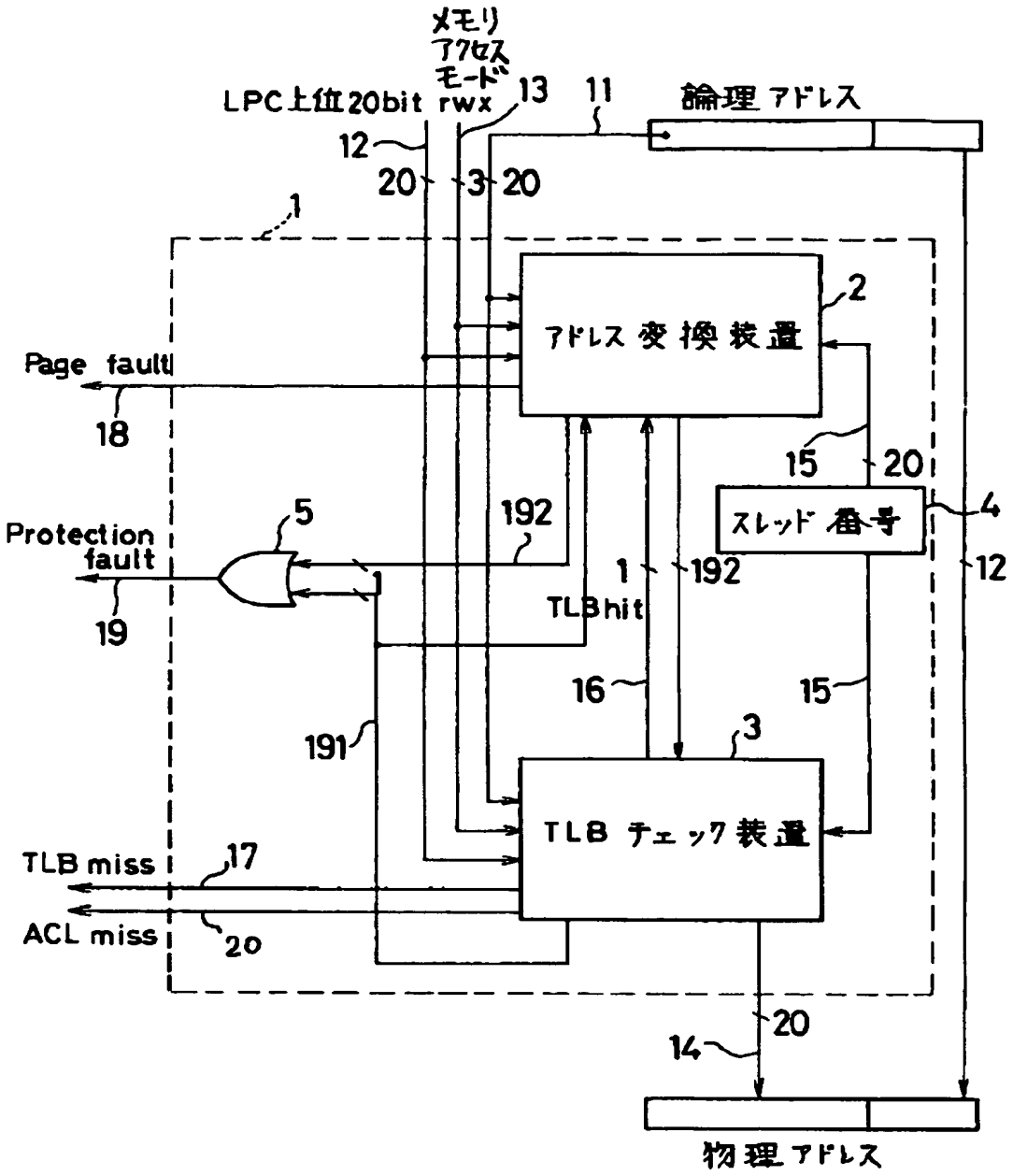
【図１１】　第４の実施例の変形例を示す概略構成図。

【図１２】　スレッド番号あるいは領域番号の全部または一部をマスクする構成の一例を示す図。

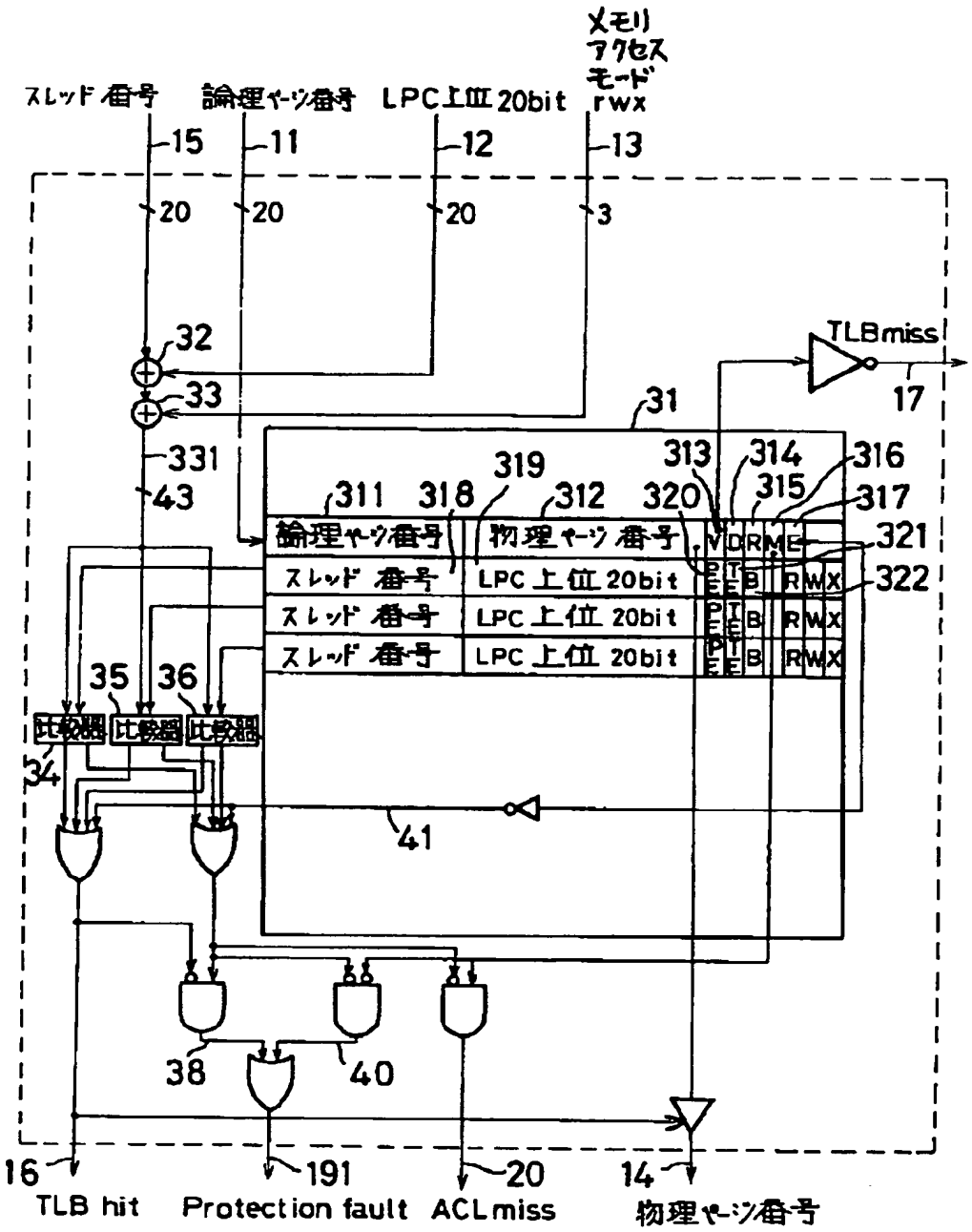
【符号の説明】

１…メモリ管理装置
２…アドレス変換装置
３…ＴＬＢチェック装置
４…スレッド番号記憶部
２１、２２、２３…アドレス変換器
２５…ルートポインタ
２１２、２３２…アドレス合成器
２１７～２１９、２３７～２３９…比較器
２２５、２４５…ＡＣＬ切替え装置
２２７、２４７…アドレス変換中止装置
３１…ＴＬＢ
３２、３３…連結器
３４、３５、３６…比較器
８０、９０、１００…論理アドレス空間
８１、８２…スレッド
８３、９４、１０４…実行しようとしている命令
８４、９５、１０５…アクセスの種類検出部
８５、９６、１０６…ページテーブル
８６…スレッド番号検出部
８７、９８…物理アドレス／プロテクション・フォルト信号出力部
８８、９９、１１０…ページフォルト検出部
９１、９２、９３、１０１、１０２、１０３…領域
９７、１０８…実行しようとしている命令が存在するアドレスを含む領域の番号検出部
１０７、１１２…ＡＣＬテーブル
１０９…物理アドレス出力部
１１１…アクセスしようとする先のアドレスを含む領域の番号検出部

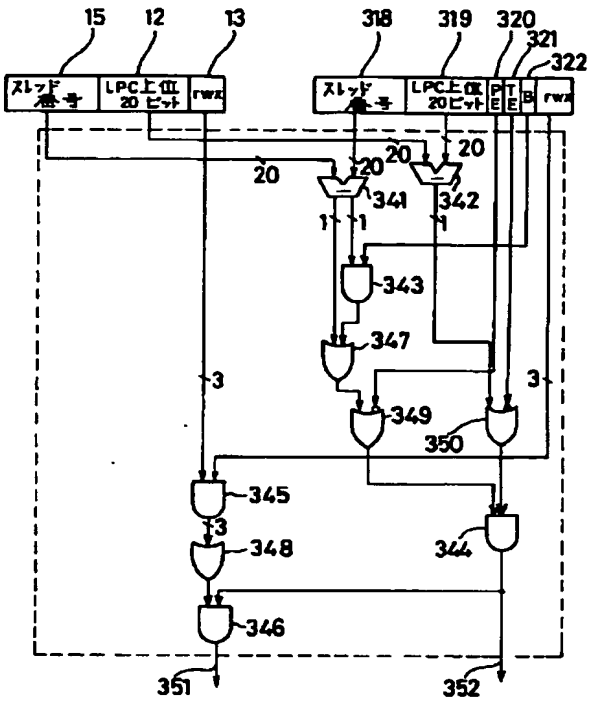
【図1】


メモリ
アクセス
モード
LPC上位20bit
rwx
13
11
論理アドレス
12
20
3
20
1
アドレス変換装置
2
Page fault
18
15
20
スレッド番号
4
5
192
Protection
fault
19
1
192
12
TLBhit
16
15
191
3
TLB チェック装置
TLB miss
17
ACL miss
20
20
14
物理アドレス

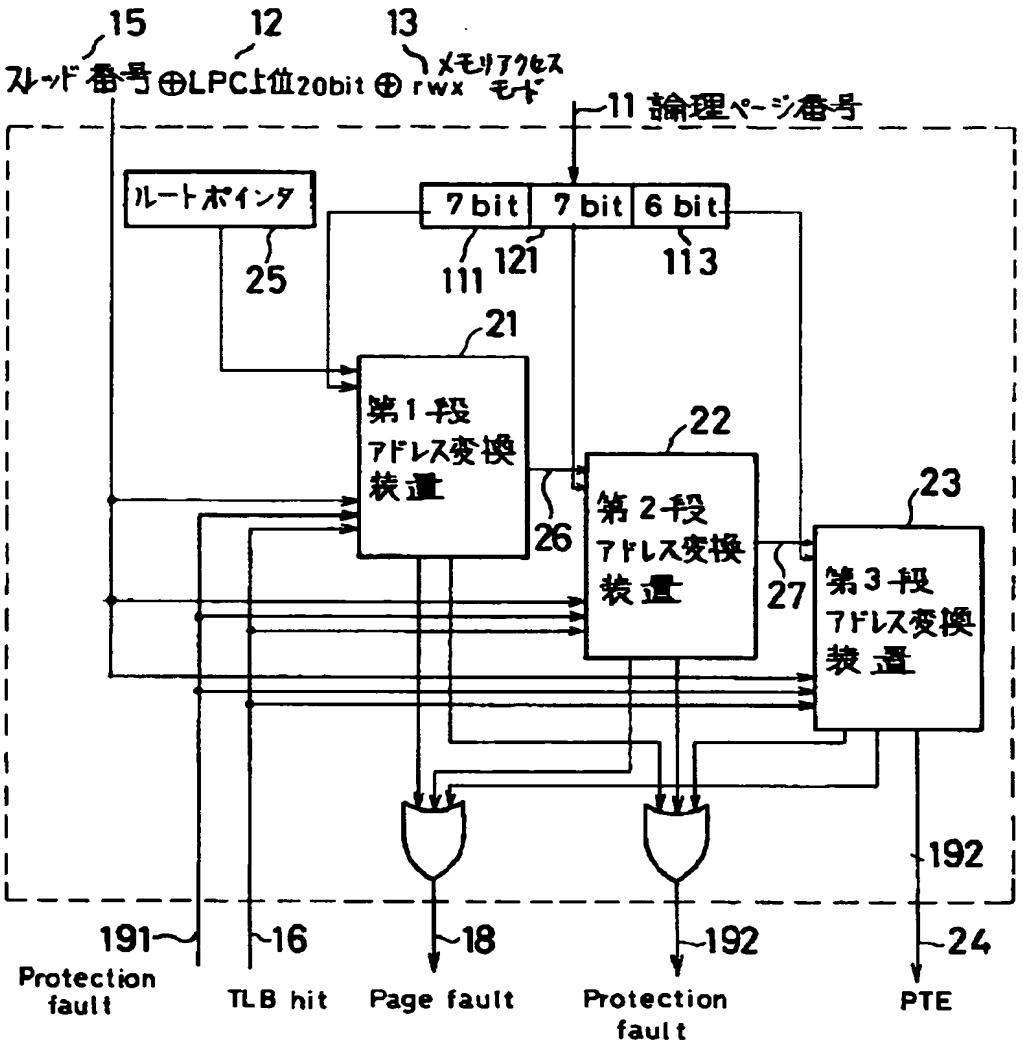
【図２】


スレッド番号
論理ページ番号
LPC上位20bit
メモリアクセスモードrwx
15
11
12
13
20
20
20
3
32
33
331
43
31
TLBmiss
17
311
318
319
312
320
313
314
315
316
317
321
322
論理ページ番号
物理ページ番号
V
D
R
M
E
スレッド番号
LPC上位20bit
PTE
B
RWX
スレッド番号
LPC上位20bit
スレッド番号
LPC上位20bit
35
36
比較器
比較器
比較器
34
41
38
40
16
TLB hit
191
Protection fault
20
ACL miss
14
物理ページ番号

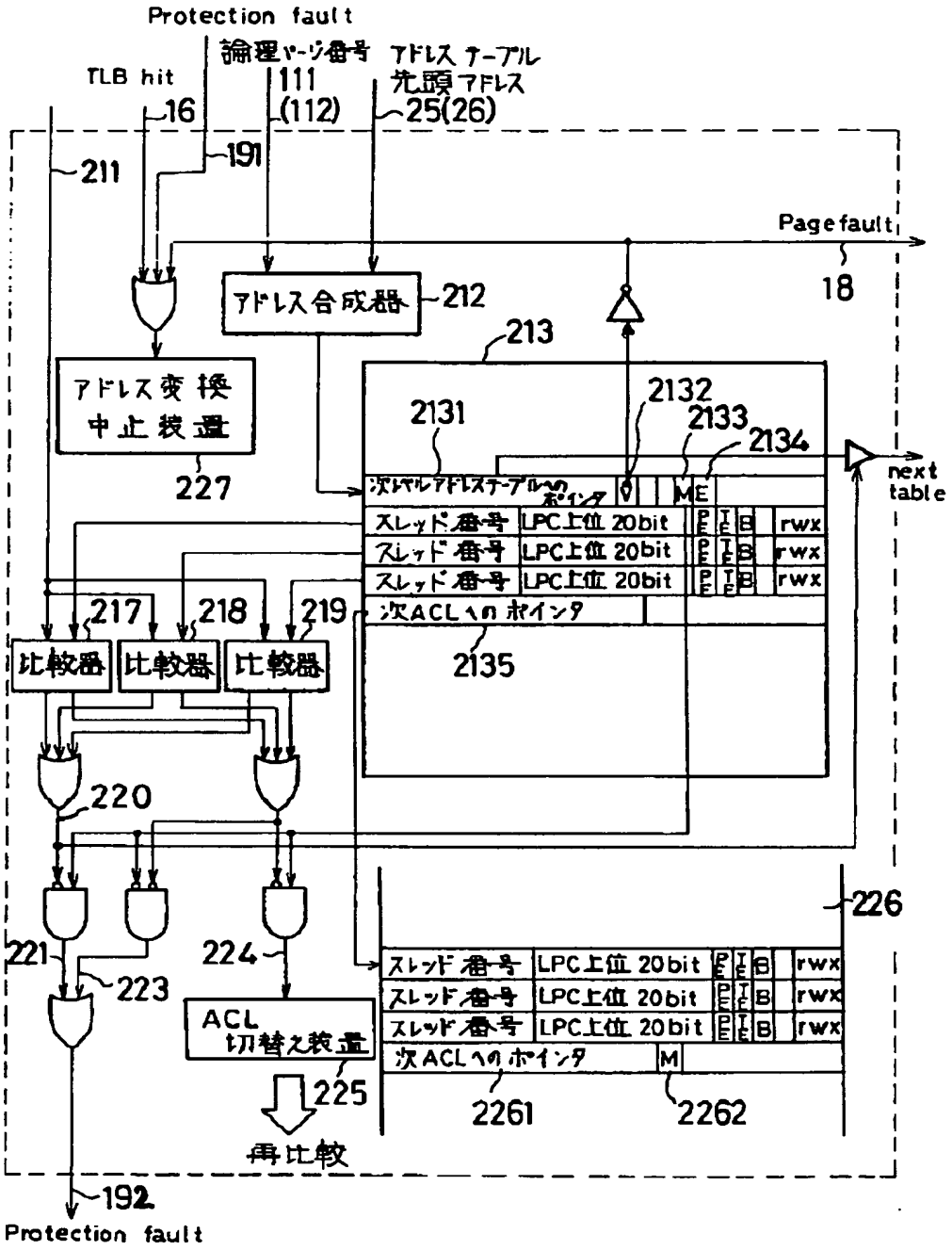
【図３】


15
12
13
318
319
320
321
322
スレッド番号
LPC上位20ビット
rwx
PTE
B
20
341
342
1
343
347
3
349
350
345
348
344
346
351
352

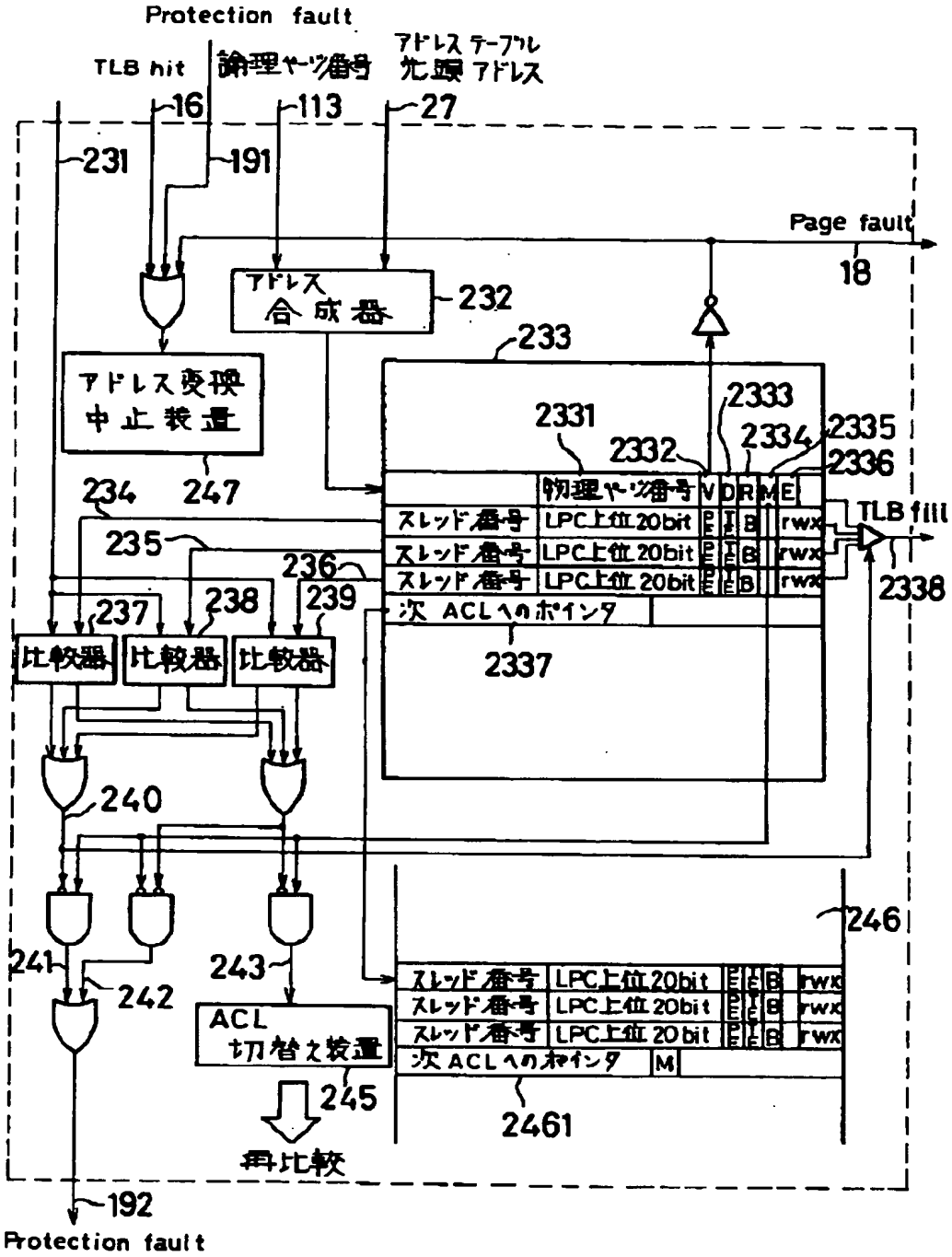
【図４】


15
12
13
スレッド番号
⊕LPC上位20bit
⊕ rwx
メモリアクセスモード
11 論理ページ番号
ルートポインタ
25
7 bit
7 bit
6 bit
111
121
113
21
第1段アドレス変換装置
22
第2段アドレス変換装置
23
第3段アドレス変換装置
26
27
192
191
16
18
192
24
Protection fault
TLB hit
Page fault
Protection fault
PTE

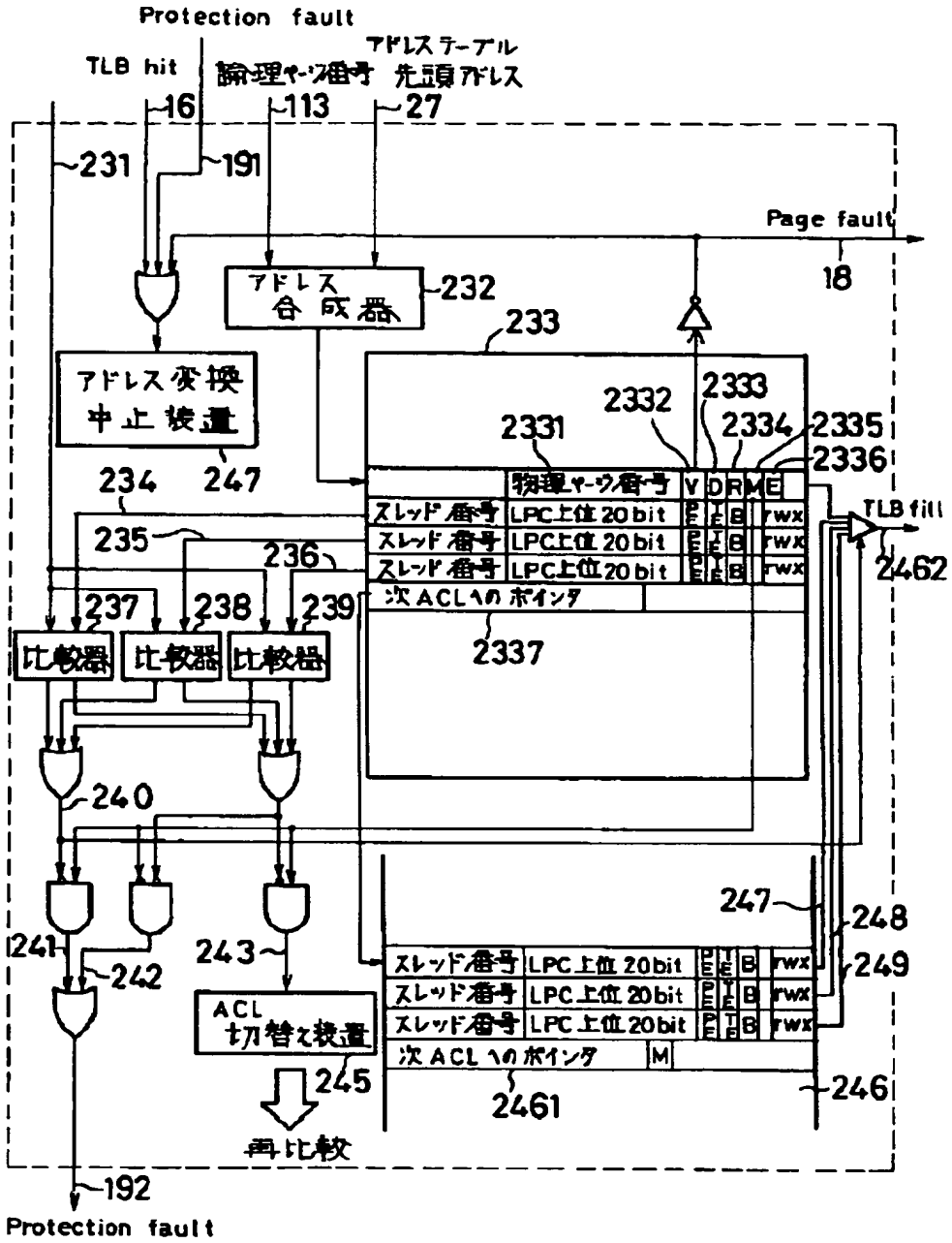
【図5】


Protection fault
論理ページ番号
111
(112)
アドレス テーブル
先頭アドレス
25(26)
TLB hit
16
191
211
Page fault
18
アドレス合成器
212
213
アドレス変換
中止装置
227
2131
2132
2133
2134
next
table
次レベルアドレステーブルへのポインタ
V
ME
スレッド番号
LPC上位 20bit
PE
IE
B
rwx
次ACLへのポインタ
2135
217
218
219
比較器
比較器
比較器
220
226
221
223
224
ACL
切替え装置
225
再比較
2261
M
2262
192
Protection fault

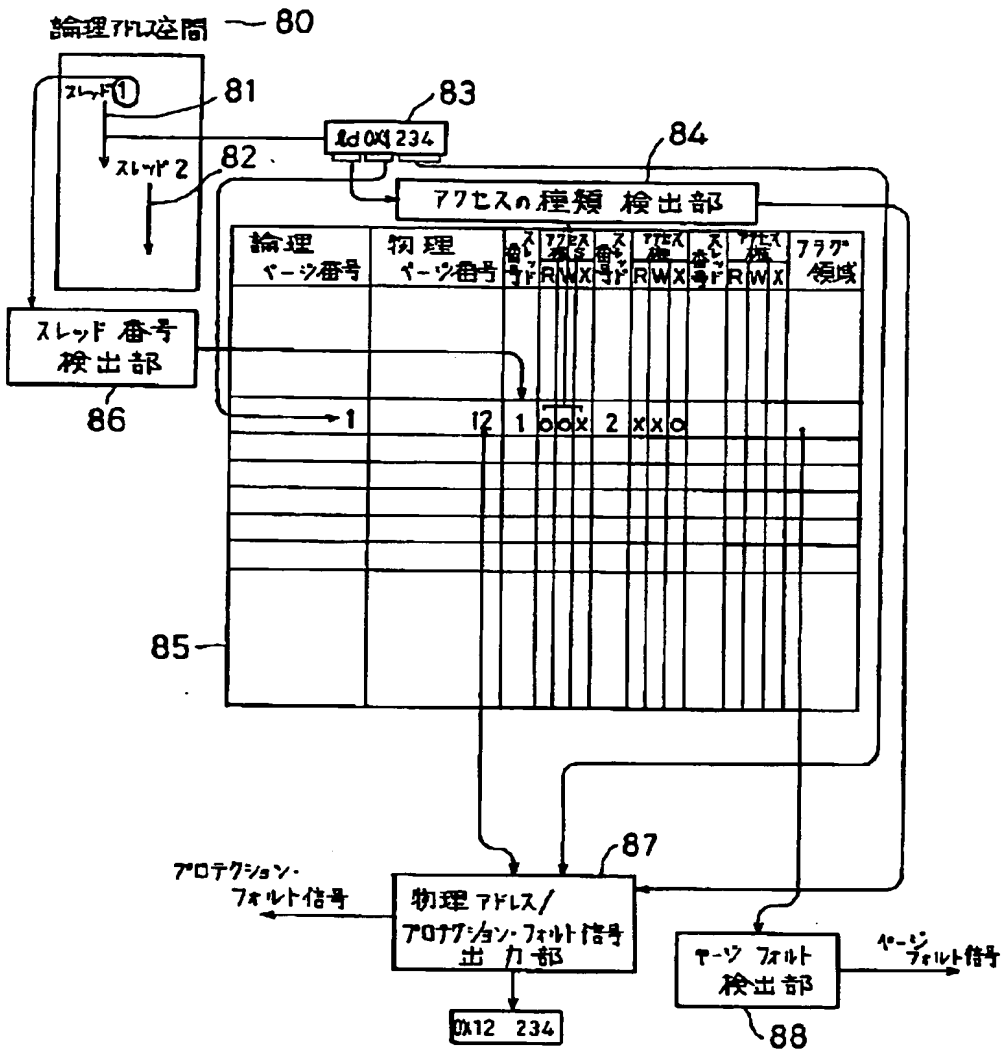
【図６】


Protection fault
TLB hit
論理ページ番号
アドレス テーブル 先頭 アドレス
16
113
27
231
191
Page fault
18
アドレス 合成器
232
233
アドレス変換 中止装置
247
2331
2332
2333
2334
2335
2336
234
物理ページ番号 V D R ME
スレッド番号 LPC上位20bit P I B rwx
TLB fill
235
236
2338
237
238
239
次 ACLへのポインタ
比較器
比較器
比較器
2337
240
246
241
242
243
スレッド番号 LPC上位20bit E E B rwx
ACL 切替え装置
245
次 ACLへのポインタ M
2461
再比較
192
Protection fault

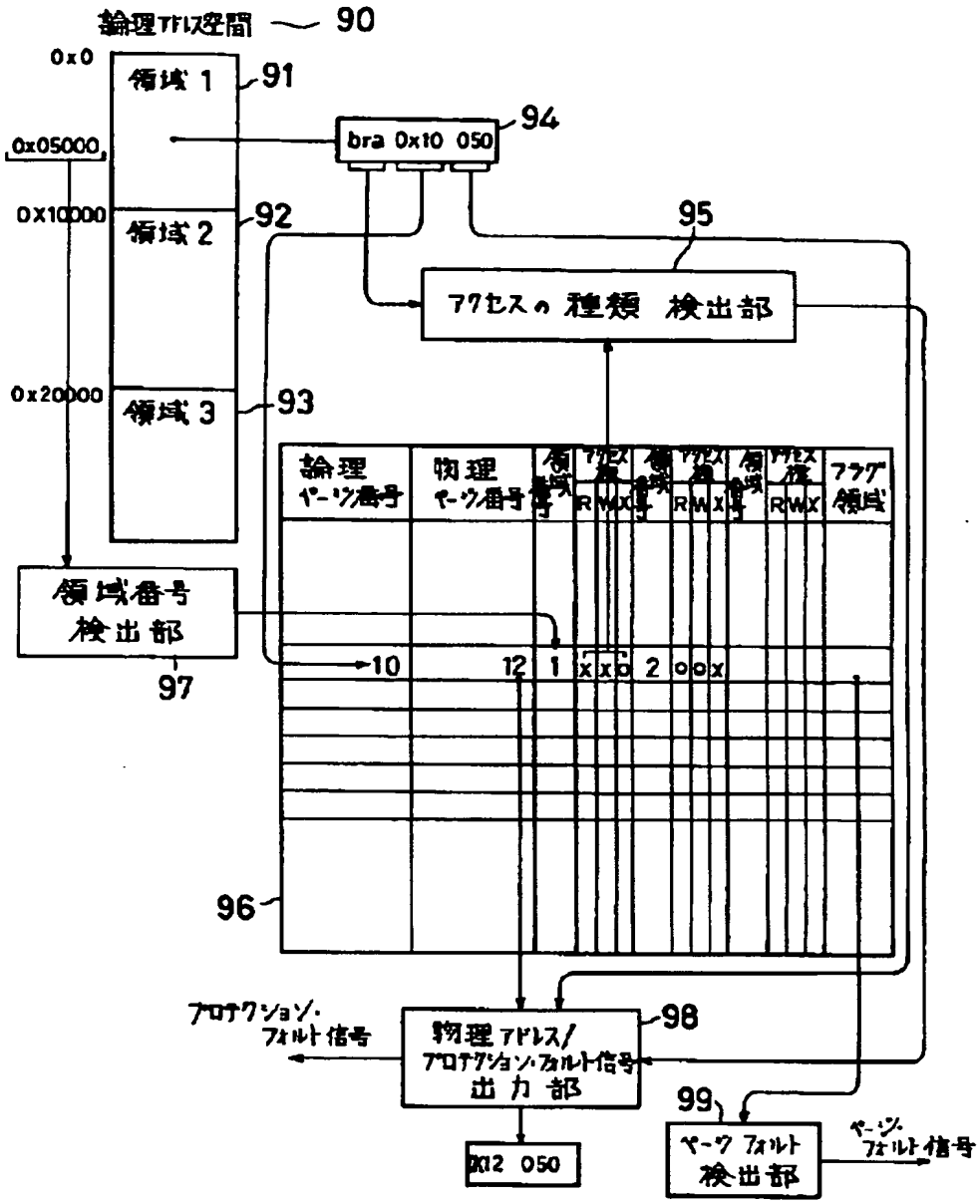
【図７】


Protection fault
TLB hit
論理ページ番号
アドレステーブル 先頭アドレス
16
113
27
231
191
Page fault
18
アドレス 合成器
232
233
アドレス変換 中止装置
247
234
2331
2332
2333
2334
2335
2336
物理ページ番号
V
D
R
M
E
スレッド番号
LPC上位20bit
rwx
235
236
237
238
239
比較器
比較器
比較器
次ACLへのポインタ
2337
TLB fill
2462
240
241
242
243
247
248
249
ACL 切替え装置
245
再比較
246
2461
M
192
Protection fault

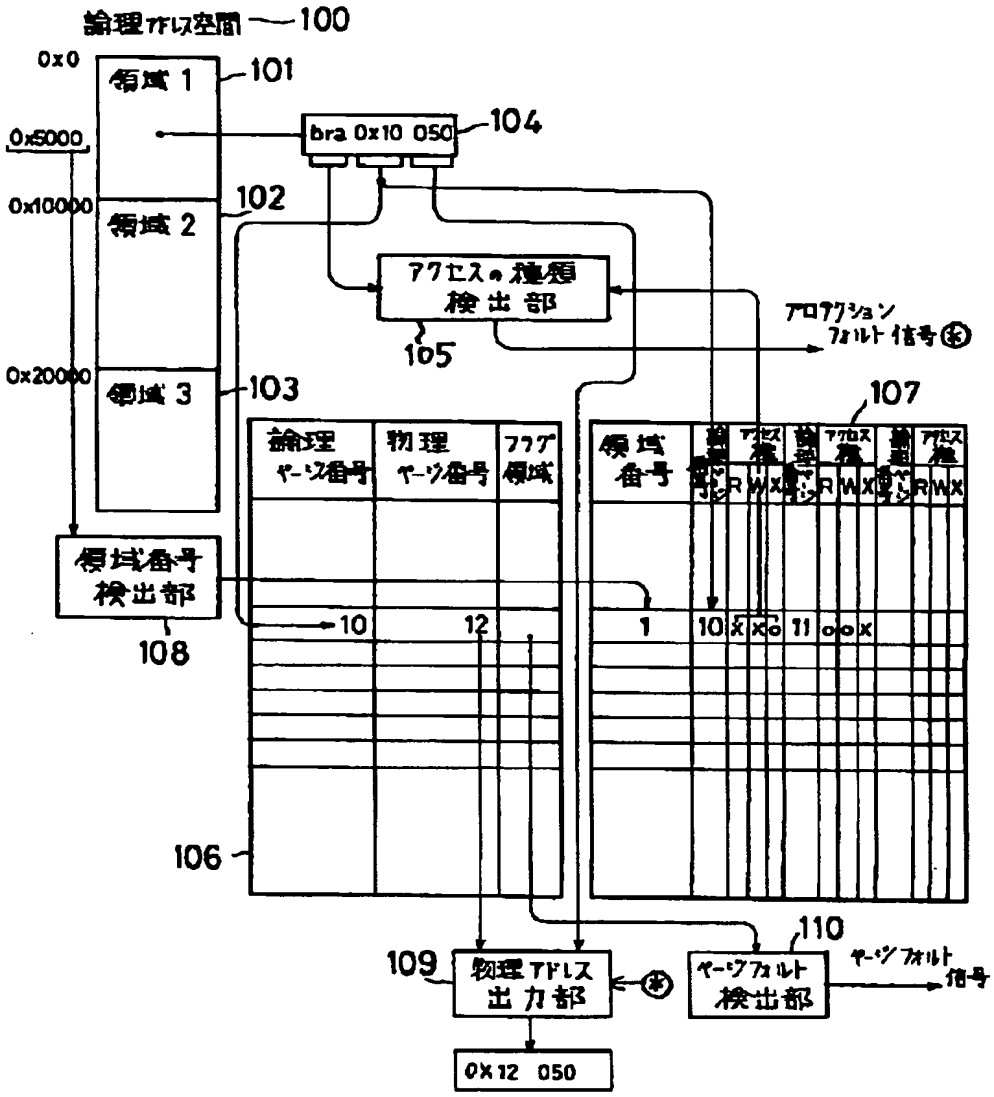
【図８】


論理アドレス空間
80
スレッド1
81
スレッド2
82
Address 234
83
84
アクセスの種類 検出部
論理 ページ番号
物理 ページ番号
スレッド番号
アクセス権
R W X
フラグ領域
スレッド番号 検出部
86
1
12
1
o o x
2
x x o
85
87
アプロテクション・フォルト信号
物理アドレス/アプロテクション・フォルト信号 出力部
0x12 234
ページ フォルト 検出部
88
ページフォルト信号

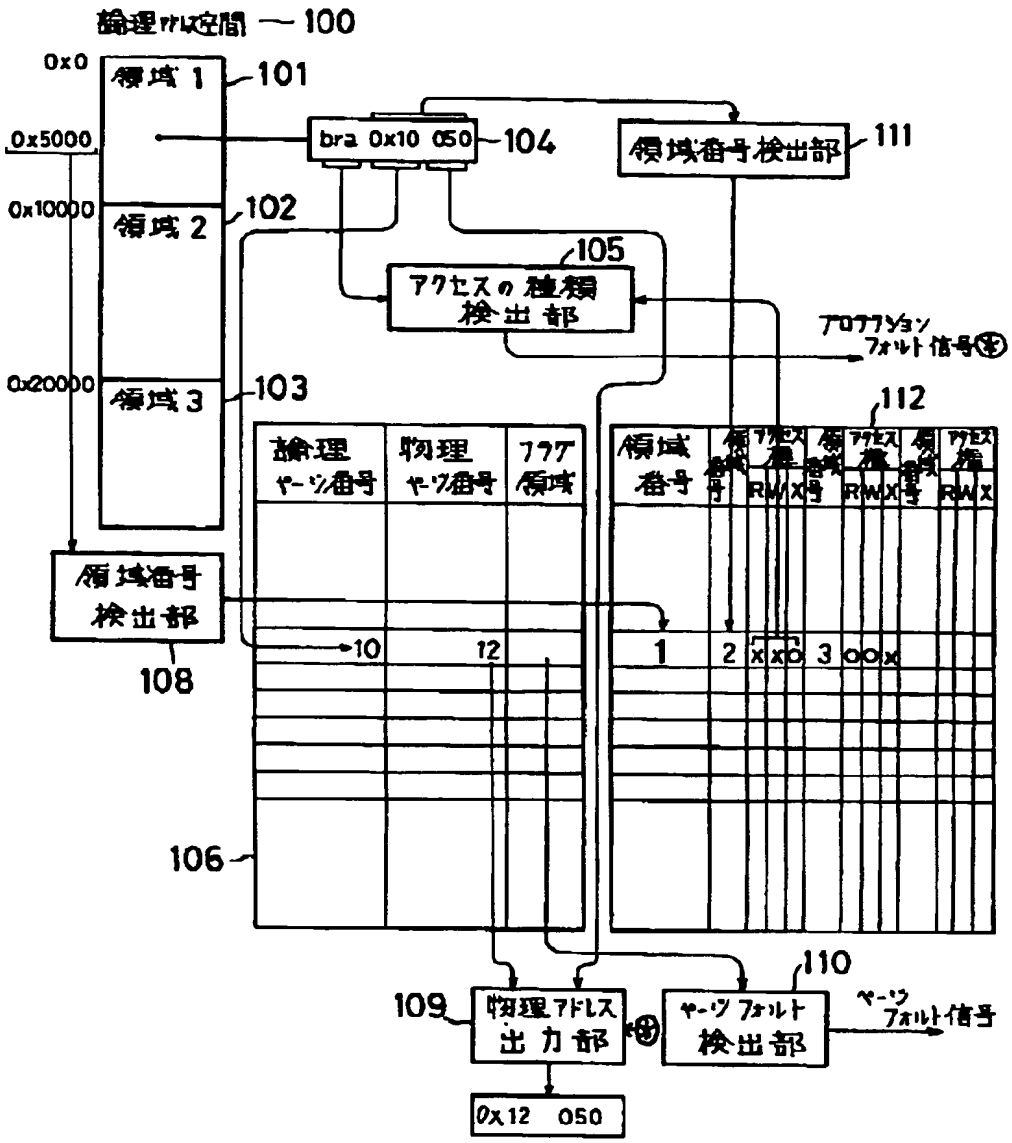
【図９】


論理アドレス空間
90
0x0
領域 1
91
0x05000
bra 0x10 050
94
0X10000
領域 2
92
95
アクセスの種類 検出部
0x20000
領域 3
93
論理 ページ番号
物理 ページ番号
領域番号
アクセス権 R W X
フラグ領域
領域番号 検出部
97
10
12
1 x x o 2 o o x
96
プロテクション・フォルト信号
物理アドレス/ プロテクション・フォルト信号 出力部
98
0X12 050
99
ページ フォルト 検出部
ページ・フォルト信号

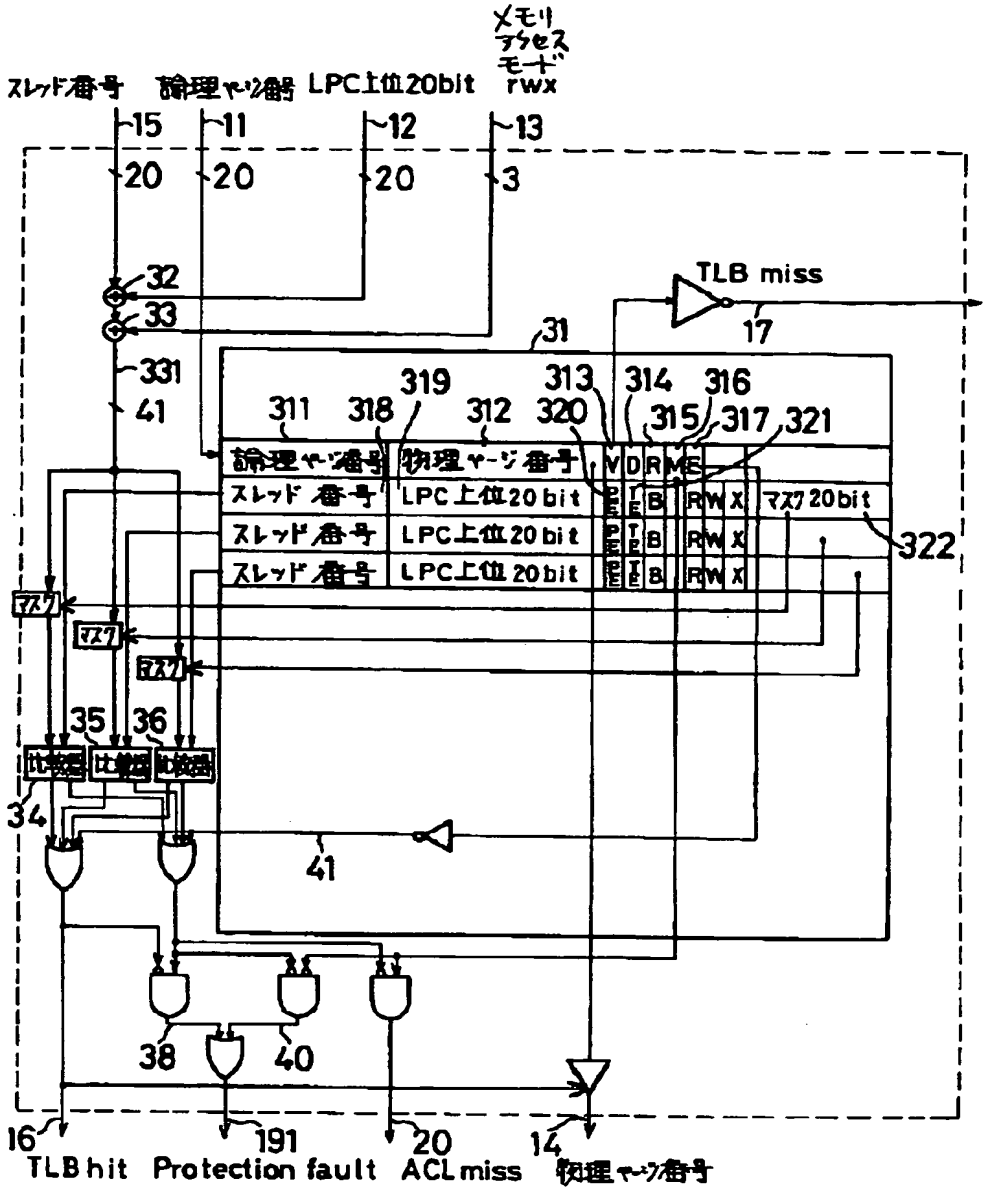
【図１０】


論理アドレス空間
100
0x0
領域 1
101
0x5000
bra 0x10 050
104
0x10000
領域 2
102
アクセスの種類
検出部
105
アクセス違反
フォルト信号
0x20000
領域 3
103
107
論理
ページ番号
物理
ページ番号
フラグ
領域
領域
番号
アクセス
権
R
W
X
10
12
1
10
11
領域番号
検出部
108
106
109
物理アドレス
出力部
110
ページフォルト
検出部
ページフォルト
信号
0x12 050

【図１１】


論理ｱﾄﾞﾚｽ空間
100
0x0
領域１
101
0x5000
0x10000
領域２
102
0x20000
領域３
103
bra 0x10 050
104
領域番号検出部
111
アクセスの種類検出部
105
ﾌﾟﾛﾃｸｼｮﾝﾌｫﾙﾄ信号
112
論理ページ番号
物理ページ番号
ﾌﾗｸﾞ領域
領域番号
領域番号
ｱｸｾｽ権
RWX
領域番号検出部
108
10
12
1
2
xxo
3
ooo
106
109
物理ｱﾄﾞﾚｽ出力部
ﾍﾟｰｼﾞﾌｫﾙﾄ検出部
110
ﾍﾟｰｼﾞﾌｫﾙﾄ信号
0x12 050

【図１２】



フロントページの続き

(72)発明者　野末　浩志
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　前田　賢一
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　瀬川　英生
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　岡本　利夫
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内